MONTANA
IDAHO
NEVADA
UTAH

Research Note

USDA FOREST SERVICE INT-273
INTERMOUNTAIN FOREST & RANGE EXPERIMENT STATION
507–25th STREET, OGDEN, UTAH 84401

August 1979

PARMS -- A Computer Program to Modify FINSYS-2 TABLE-2 and Other FORTRAN Programs

Terrence Throssell[1]

ABSTRACT

Presents a method for reducing the core size requirements of FINSYS-2 TABLE-2 and other FORTRAN programs. Describes the use of a preprocessor to get information about the size requirements of a particular job and how to incorporate this information into producing a tailor made executable program.

KEYWORDS: computers, FORTRAN, FINSYS-2 TABLE-2, dimension modification

INTRODUCTION

In the late 1960's a standard version of the FORTRAN program TABLE was published as a portion of the general inventory data reduction system FINSYS[2]. Documentation explained how to modify the dimensions of the standard version to meet each user's special applications.

---

[1]Mathematician located at the Renewable Resources Evaluation Project, Intermountain Forest and Range Experiment Station, U.S. Department of Agriculture, Ogden, Utah 84401.

[2]Wilson, R. W., and R. C. Peters. 1967. The Northeastern Forest Inventory Data Processing System, USDA For. Serv. Res. Pap. NE-61 and NE-70 to 78. Northeastern For. Exp. Stn., Broomall, Pa.

It soon became apparent, however, that most users often needed to modify the program dimensions. Some users enlarged the dimensions to increase the work capabilities of the program while other users cut back its size to run on computers with limited storage. Experienced users spent much time calculating required dimension limits and modifying the program accordingly. Inexperienced users made frustrating mistakes.

Then, in the early 1970's, a solution was proposed in the form of a computer program, PARMS. This program has since been refined to run with the latest version of FINSYS-2 TABLE-2 being used by the Forest Service and cooperators on present generation computers. PARMS eliminates the need for programers to make dimension modifications in the FORTRAN program FINSYS-2 TABLE-2. PARMS not only saves programer time, but also utilizes computer storage more efficiently. Futhermore, the general method used by PARMS can be applied to other FORTRAN programs.

PARMS METHOD

The general PARMS method has two requirements: (1) the FORTRAN program must be structured so that all dimension changes can be made in the main calling routine, and (2) the main calling routine must be short, 25 lines or less.

Most FORTRAN programs already meet the first requirements; few meet the second. But generally little work is required to split a long main routine into a shorter main routine with a long control subroutine. If these two requirements are met, it becomes very easy to modify dimensioned space in a FORTRAN program. One need only replace the main routine with another written to the required dimension limits.

The rewriting and replacing of the main routine can either be done manually by a programer or it can be done automatically by the computer. Consider the simple main routine in figure 1. To modify the dimension limits of arrays A, B, and C from 5000, 10000, and 25000 to 3000, 5000, and 2000 as in figure 2, only six numbers need to be changed. These changes can easily be made by the computer. Figure 3 shows a computer program, the method used by PARMS, to accomplish the task. The program reads the dimension limits punched on a data card and writes the new FORTRAN main routine on logical unit 8, a scratch file. Job control language handles the task of replacing the old main routine with the new one.

Job control language varies greatly from one computer to another. Figure 4 shows the processing flow for a regular FORTRAN program. Figure 5 shows the same program setup under the PARMS method. In each diagram, dotted lines show that portion of the processing that is transparent to the user. To the user the only difference in the PARMS method is that it does not require the input main FORTRAN routine but does require input dimension limits.

In review, PARMS can be applied to FORTRAN programs with short main calling routines structured to facilitate dimension modification. A program and associated job control language can be written to automate the rewriting and replacing of this main routine with another routine written to the desired dimension limits.

```
      DIMENSION A( 5000),B(10000),C(25000)
      DATA I,J,K/ 5000,10000,25000/
      CALL CONTRL(A,B,C,I,J,K)
      STOP
      END
```

Figure 1.--Sample FORTRAN main routine

```
      DIMENSION A( 3000),B( 5000),C( 2000)
      DATA I,J,K/ 3000, 5000, 2000/
      CALL CONTRL(A,B,C,I,J,K)
      STOP
      END
```

Figure 2.--Same routine as figure 1 but with modified dimension limits

```
      DATA LU5,LUOUT/ 5,8/
      READ(LU5,5) I,J,K
    5 FORMAT(3I5)
      REWIND LUOUT
      WRITE(LUOUT,10) I,J,K
   10 FORMAT(6X,'DIMENSION A(', I5, '),B(', I5, '),C(', I5, ')' )
      WRITE(LUOUT,20) I,J,K
   20 FORMAT(6X,'DATA I,J,K/', I5, ',', I5, ',', I5, '/' )
      WRITE(LUOUT,30)
   30 FORMAT(6X,'CALL CONTRL(A,B,C,I,J,K)',/,6X,'STOP',/,6X,'END')
      END FILE LUOUT
      REWIND LUOUT
      STOP
      END
```

Figure 3.--FORTRAN program to write the main routines shown in figures 1 and 2

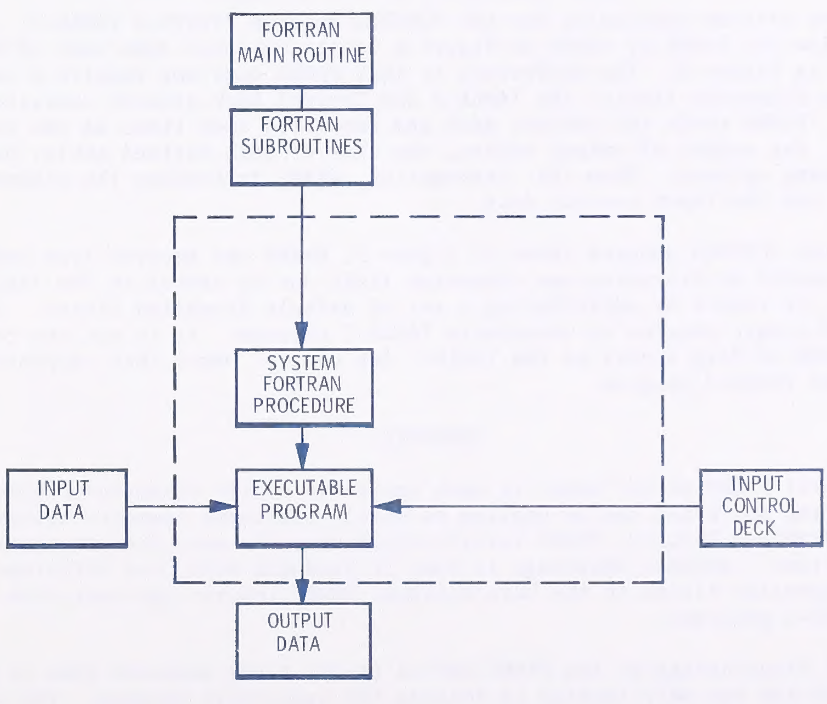

Figure 4.--Regular FORTRAN program processing flow.

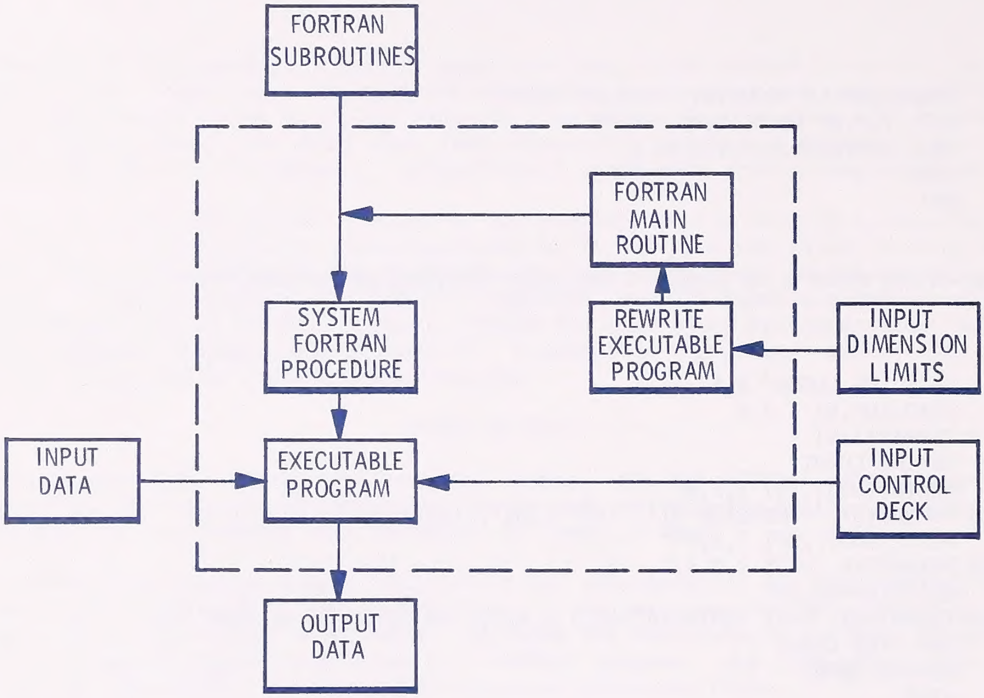

Figure 5.--Generalized method processing flow.

PARMS AND FINSYS-2 TABLE-2

PARMS was written especially for the FORTRAN program FINSYS-2 TABLE-2. The processing flow for PARMS as shown in figure 6 varies slightly from that of the general method shown in figure 5. The difference is that PARMS does not require a separate input for the dimension limits; the TABLE-2 Job Control Deck already contains this information. PARMS reads the control deck and tabulates such items as the number of input tables, the number of output tables, the size of each defined table, plus various other processing options. From this information, PARMS determines the dimension limits necessary to run the input control deck.

Unlike the FORTRAN program shown in figure 3, PARMS can recover from bad input. If PARMS is unable to determine any dimension limit due to errors in the TABLE-2 Job Control Deck, it reacts by substituting a set of default dimension limits. In this manner, PARMS always creates an executable TABLE-2 program. It is not the responsibility of PARMS to flag errors in the TABLE-2 Job Control Deck; that responsibility rests with the TABLE-2 program.

COMMENTS

The general PARMS method makes it much easier to modify dimensioned space in a FORTRAN program, and PARMS can be applied to widely different computer systems. In the case of FINSYS-2 TABLE-2, PARMS totally eliminates the need for programers to make any modifications. Another advantage is that it produces more core efficient programs. By cutting dimension limits to the bare minimum, PARMS insures the most core efficient FINSYS-2 TABLE-2 programs.

A slight disadvantage to the PARMS method is the extra computer time it takes to create and use the new main routine in forming the executable program. The cost of this computer time, however, generally is insignificant compared to a programer's time.

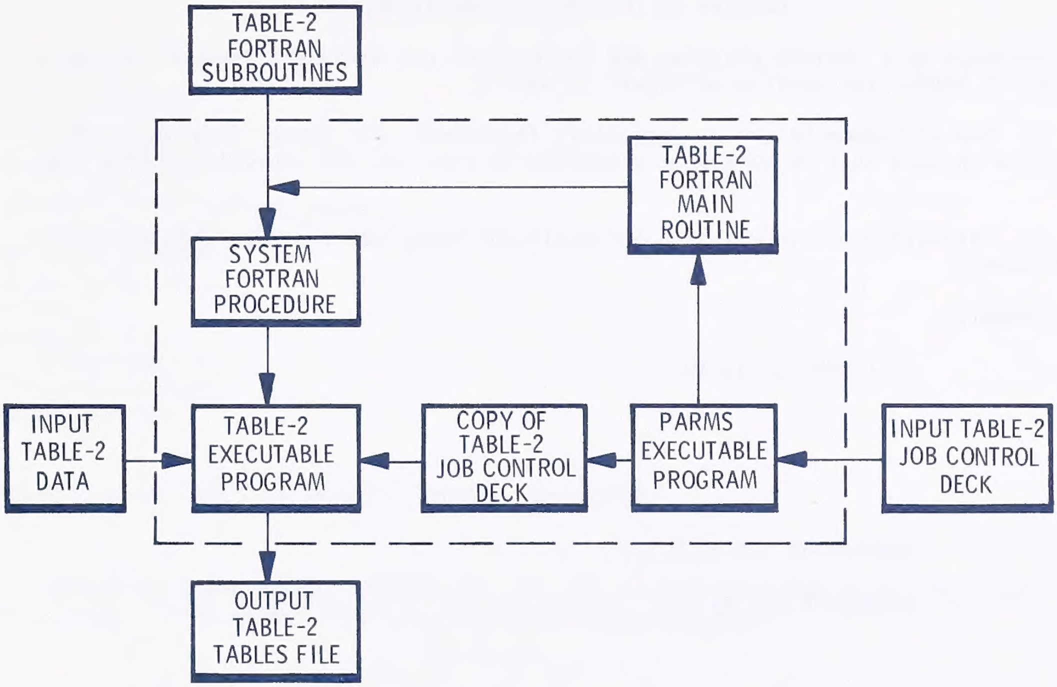

Figure 6.--PARMS processing flow.

APPENDIX A

Caution on FORTRAN Program Structure

Although most FORTRAN programs and subroutines are already structured to facilitate the use of PARMS, two caution notes are in order.

1. Use of common blocks is explicitly forbidden. All arrays that may have dimension changes must be passed as arguments in calls to the subroutines that use them.

2. Variable dimensioning must be consistent among all routines referencing a given array.

Example:

```
        CALL SUB1 (X,10,10)
        .
        .
        .
        SUBROUTINE SUB1(X,M,N,)
        DIMENSION X(M,N)
        .
        .
        .
        CALL SUB2(X,5,10)
```

The above call would cause disastrous realignment of the array X.

APPENDIX B

Job Control Language Examples.

Examples of the job control language required for PARMS are shown for the CDC CYBER, IBM, and UNIVAC computers. All examples assume that the FORTRAN program PARMS is stored in executable form, and that the FINSYS-2 TABLE-2 program is stored in either source, relocatable, or executable form, on the given host computer.

An explanation of the 14 lines of figure 7:

```
 1:GET,PARMS.
 2:GET,TAPE60=T1.
 3:PARMS.
 4:GET,TABSUBS.
 5:GET,CALCUL=T2.
 6:REWIND,TAPE8,TABSUBS,CALCUL,Z.
 7:COPYCR,TAPE8,Z.
 8:COPYCR,TABSUBS,Z.
 9:COPYCR,CALCUL,Z.
10:PACK,Z.
11:FTN,I=Z,L=0.
12:ATTACH,TAPE1=T3.
13:DEFINE,TAPE2=T4.
14:LGO(,OUT995).
```

Figure 7.--CDC CYBER job control language for PARMS

The files named PARMS, TABSUBS, T1, T2, T3, and T4 must already be catalogued on the system. Files TAPE8, TAPE9, Z, and OUT995 are temporary.

FILE NAME	DESCRIPTION
PARMS	EXECUTABLE PROGRAM PARMS
TABSUBS	SOURCE FOR TABLE-2 SUBROUTINES
T1	TABLE-2 JOB CONTROL DECK
T2	SOURCE FOR SUBROUTINE CALCUL
T3	INPUT TABLE-2 DATA
T4	OUTPUT TABLE-2 TABLES FILE

Lines 1-3 — Execute program PARMS. Read the TABLE-2 Job Control Deck from unit 60 and copy it to temporary unit 9 (not shown). Write the new main routine on temporary unit 8.

Lines 4-10 — Build a temporary source file of the new main routine, the new subroutine CALCUL, and the standard TABLE-2 subroutines.

Line 11 — Compile the TABLE-2 program from the temporary new source file.

Lines 12-14 Create and execute the new temporary TABLE-2 program. Read the Job Control Deck from temporary unit 9 (not shown). Read the input data from unit 1 and write the output tables file on unit 2. Direct the print to the temporary file named OUT995 for later disposition.

An explanation of the 28 lines of figure 8:

```
 1://CALCUL   EXEC PGM=COPY
 2://STEPLIB  DD DSNAME=FST.PGMLIB,DISP=SHR
 3://FT02F001 DD DSNAME=&&CALCUL,UNIT=WORK,SPACE=(80,(1000,100)),          X
 4://            DISP=(NEW,PASS),DCB=BLKSIZE=80
 5://FT01F001 DD *
 6:     TABLE-2 USER MODIFIABLE FORTRAN SUBROUTINE CALCUL
 7:/*
 8://PARMS    EXEC PGM=PARMS
 9://STEPLIB  DD DSNAME=FST.PGMLIB,DISP=SHR
10://FT06F001 DD SYSOUT=E
11://FT08F001 DD DSNAME=&&MAIN,UNIT=WORK,SPACE=(80,(100,100)),             X
12://            DISP=(NEW,PASS),DCB=BLKSIZE=80
13://FT09F001 DD DSNAME=&&DECK,UNIT=WORK,SPACE=(80,(1000,100)),            X
14://            DISP=(NEW,PASS),DCB=BLKSIZE=80
15://FT05F001 DD *
16:     TABLE-2 JOB CONTROL DECK
17:/*
18://TABLE    EXEC FORTGCLG
19://FORT.SYSIN DD DSNAME=&&MAIN,DISP=(OLD,DELETE)
20://           DD DSNAME=&&CALCUL,DISP=(OLD,DELETE)
21://LKED.OLDLOAD DD DSNAME=FST.PGMLIB,DISP=SHR
22://LKED.SYSIN DD *
23: INCLUDE OLDLOAD(TABLE)
24: ENTRY MAIN
25:/*
26://GO.FT05F001 DD DSNAME=&&DECK,DISP=(OLD,DELETE)
27://FT01F001 DD (INPUT TABLE-2 DATA FILE DEFINITION)
28://FT02F001 DD (OUTPUT TABLE-2 TABLES FILE DEFINITION)
```

Figure 8.--IBM job control language for PARMS

Executable programs COPY, PARMS, and TABLE must be catalogued on the system. Data sets &&MAIN, &&DECK, and &&CALCUL are temporary.

Lines 1-7 Copy FORTRAN source statements for subroutine CALCUL to a temporary file for later use in compiling the TABLE-2 program.

Lines 8-17 Execute program PARMS. Read the TABLE-2 Job Control Deck as card input. Copy the control deck to the temporary unit 9. Write the new main routine on temporary unit 8.

Lines 18-20 Compile the new main routine and new subroutine CALCUL.

Lines 21-25 Create the executable TABLE-2 program from the new object main routine, the new object subroutine CALCUL, and the old load module of the standard TABLE-2 program stored in program library FST.PGMLIB.

Lines 26-28 Execute the new TABLE-2 program. Read the control deck from the copied temporary file. Read the input data from unit 1. Write the output tables file on unit 2.

An explanation of the 21 lines of figure 9:

```
 1:@FOR,IS CALCUL
 2:      TABLE-2 USER MODIFIABLE FORTRAN SUBROUTINE CALCUL
 3:@ASG,T 3.
 4:@ASG,T 4.
 5:@ASG,T 9.
 6:@XQT PARMS
 7:      TABLE-2 JOB CONTROL DECK
 8:@EOF
 9:@FOR,IS MAIN
10:      TABLE-2 FORTRAN MAIN ROUTINE BEGINNING STATEMENTS
11:@ADD 4.
12:      TABLE-2 FORTRAN MAIN ROUTINE ENDING STATEMENTS
13:@MAP,INX     ,TPF$.TABLE
14: IN TPF$.MAIN,.CALCUL
15: IN TABLIB.SUB1,.SUB2,.VARIAN
16:@ADD 3.
17:@ASG,OPTIONS NAME1.   . (INPUT TABLE-2 DATA FILE)
18:@USE 10.,NAME1.
19:@ASG,OPTIONS NAME2.   . (OUTPUT TABLE-2 TABLES FILE)
20:@USE 2.,NAME2.
21:@XQT TABLE
```

Figure 9.--UNIVAC job control language for PARMS

Executable program PARMS and the relocatable subroutine for TABLE-2 must be catalogued on the system.

Lines 1-2 Compile subroutine CALCUL.

Lines 3-8 Assign files for and execute program PARMS. Read the Table-2 Job Control Deck from the card reader and write the copy on temporary unit 9. Write the dimension parameters for the new main routine on temporary unit 4. Write the extended storage map collector statements on temporary unit 3.

Lines 9-12 Compile the new main routine using the dimension parameters passed from PARMS on unit 4.

Lines 13-16 Create the executable TABLE-2 program from the new relocatable main routine, the new relocatable CALCUL subroutine, the old relocatable subroutines found in the program library TABLIB, and the extended storage map collector statements passed from PARMS on unit 4.

Lines 17-21 Execute the TABLE-2 Program. Read the Job Control Deck from unit 9. Read the input data from unit 10. Write the output tables file on unit 2.

APPENDIX C

Program PARMS Source Decks

The program consists of the main routine PARMS and subroutines PDPMAP and TABFOR. Subroutine PDPMAP is used only by the UNIVAC version. Subroutine TABFOR is used by the CDC and IBM versions and can be used in place of subroutine PDPMAP for the UNIVAC version.

The value of ISYS will vary with specific machines: ISYS must be set at 0 for the IBM, 1 for the UNIVAC, and 2 for the CDC. For the CDC version, the end-of-file read check must be modified.

PARMS source decks and listings are available from the Intermountain Forest and Range Experiment Station, Ogden, Utah.

Headquarters for the Intermountain Forest and Range Experiment Station are in Ogden, Utah. Field programs and research work units are maintained in:

- Billings, Montana
- Boise, Idaho
- Bozeman, Montana (in cooperation with Montana State University)
- Logan, Utah (in cooperation with Utah State University)
- Missoula, Montana (in cooperation with University of Montana)
- Moscow, Idaho (in cooperation with the University of Idaho)
- Provo, Utah (in cooperation with Brigham Young University)
- Reno, Nevada (in cooperation with the University of Nevada)

☆ U.S. GOVERNMENT PRINTING OFFICE: 1979-0-677-019/36